



Object Level Replication

Koen Holtman

Caltech/CMS

PPDG meeting, Argonne
July 13-14, 2000

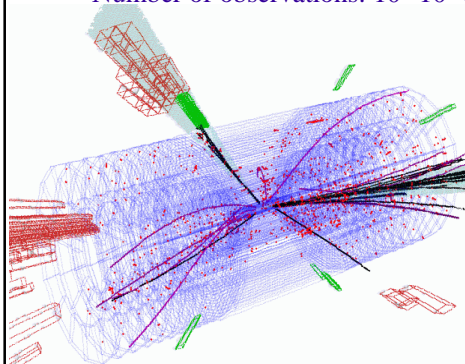
1



Scientific data analysis

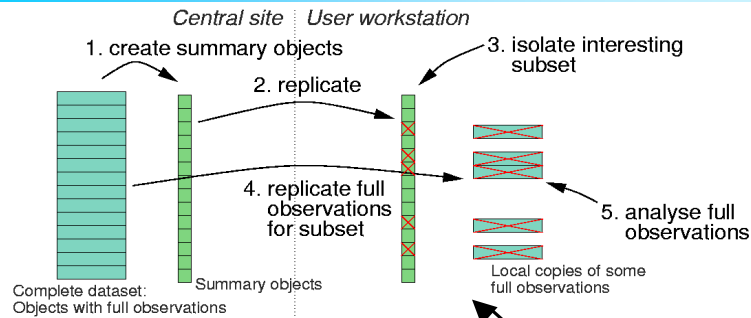


- Statistical analysis of large set of observations
 - Physics: events
 - Astronomy: stars, galaxies, and their spectra
- Individual observation (object) size: 1 KB -1 MB
- Number of observations: 10^8 - 10^9 (PBs - TBs)



Redshift 5.0 Quasar
age credit: SDSS Collaboration)

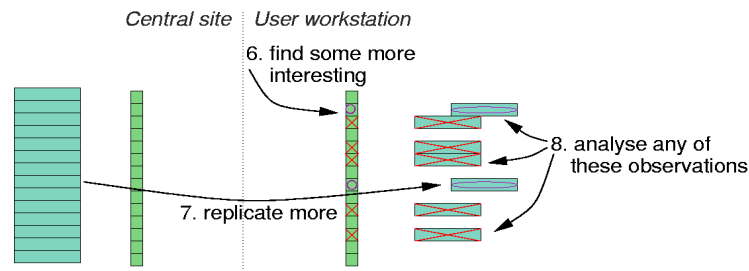
Most basic use case



- Want to build tool that supports this
- Why replicate at all?
 - More control over data availability/lifetime/stability
 - Can get more CPU power, I/O power at new location
 - Preserve sequential access patterns to data
- None of these really require replication to *local* machine

3

More complex use case



- This is where tool support (and strong indexing) really starts to become useful!
- Building tool that supports at least this use case
 - First demo around Oct/Nov 2000 (ACAT 2000, SC 2000)
 - Using physics data of the CMS experiment
 - ORCA trigger study data, at least a few GB
 - At least tags (~100 byte objects) + AODs (~10KB, maybe fake contents)

4



Data grid use case



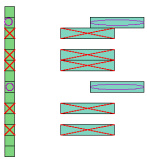
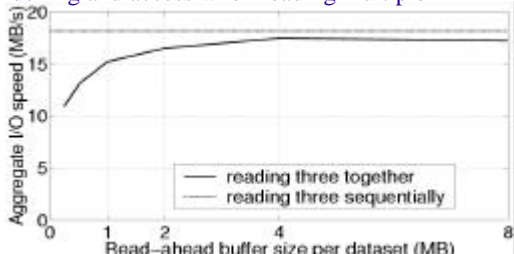
- Still more complex use case: Data Grid like:
 - Many users (100+), many sites (100+)
 - Many types of objects & versions (1000+)
 - Location of data is invisible to users because it is irrelevant to users
 - Dynamic, transparent replication, integration with job scheduling, job migration
- Want to expand into covering parts of the data grid use case, all with object-level data granularity

5



Software technology



- Existing technology:
 - Fast object-level indexing and access when reading multiple object sets together
- 
- 
- | Read-ahead buffer size per dataset (MB) | reading three together (MB/s) | reading three sequentially (MB/s) |
|---|-------------------------------|-----------------------------------|
| 0 | 10 | 18 |
| 1 | 15 | 18 |
| 2 | 16 | 18 |
| 4 | 17 | 18 |
| 8 | 18 | 18 |
- Re-packing objects into (database) files
 - Fast FTP for these files
 - Main open (research) questions:
 - How to do scheduling, resource allocation
 - Approach: extend algorithms that work on files to work on (possibly overlapping) object sets
 - Data model, metadata model?

6



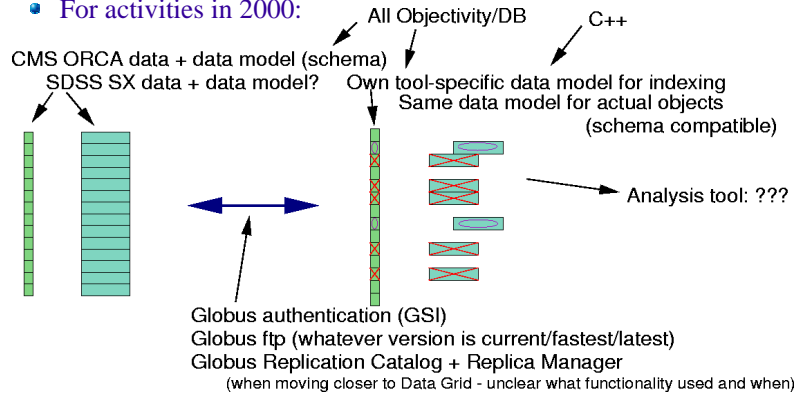
Practical questions



- Practical questions at least as complicated as research questions!

- Which data, software platforms to use/integrate with

- For activities in 2000:



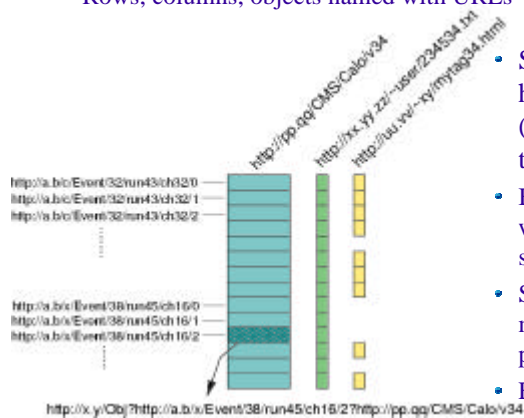
7



Naming objects 1



- My current thoughts on naming
- Physics objects are organised as a sparse (SQL) Table
 - Rows=events, columns=physics objects
 - Rows, columns, objects named with URLs



- Software tools know how to resolve (combinations of) URLs to physics data
- Bonus: paste of URL into web browser may give something useful
- Some regularity in URLs makes fast lookups possible
- Event set = set of URLs!

8



Naming objects 2



- How do names (of columns) get created?
 - Everybody can create names: the goal is to have a single coherent 'object space' and prevent 'islands of information'
- Use case scenario
 - I invent a new tag
 - I Create a new **name**: **URL** pointing to metadata document in my web space (<http://home.cern.ch/~kholtman/tags/tag55.txt>)
 - My local tool can now resolve this URL to pick up a **data location record** that points to the tag data I created
 - Later I want to **publish** this tag data
 - I use a tool to register the name (URL) with **central resolver service**, at least one data location record is bound to this URL
 - Central resolver service maintains a mapping
URL -> set of (data location record)
 - This mapping is updated if replication happens

```
Name=http://home.cern.ch/~kholtman/tags/tag55.txt
type=column_of_objects
location_type=orca_3.1.0_federation
location_federation=cms1.cern.ch::MYFD
location_objectname=mydata/tag55
policy=private,readwrite
```